Association for Information Systems
AIS Electronic Library (AISeL)

MWAIS 2006 Proceedings | Midwest (MWAIS)

December 2006

Management of Web Application Development: A case study in the tenets of Web 2.0

Robert Adams
Grand Valley State University

John Reynolds
Grand Valley State University

Panel

Management of Web Application Development: A Case-Study in the Tenets of Web 2.0

D. Robert Adams
School of Computing and Information Systems
Grand Valley State University
adams@cis.gvsu.edu

John H. Reynolds
School of Computing and Information Systems
Grand Valley State University
john.reynolds@cis.gvsu.edu

ABSTRACT

Web 2.0 provides a set of tenets that describe both a business model and a technical strategy for organizations that wish to provide a flexible user interface via the Web. Many of these tenets contradict what has become common practice in the industry. However, as the use of the Web as a development platform increases, academics need to be aware of this newly emerging methodology. This paper briefly describes Web 2.0, and our experience with its principles and practices within a web-based system for curricular assessment. Specifically, this paper provides a case-study of our work on an assessment tool for higher education that started out as a traditional systems development project, and quickly evolved into a Web 2.0 project. We have two goals in this paper. Our primary purpose is to inform the reader about Web 2.0, additionally, we illustrate how Web 2.0 was successfully adopted in our project. Our aim is that others might be equally motivated by what Web 2.0 has to offer.

Keywords

System development, Web 2.0, web applications.

INTRODUCTION

Assessing curricula in higher education is becoming more important as more and more accreditation bodies begin to focus on student learning outcomes and continuous improvement of programs. One way of assessing curricula is by using exit exams given to graduating seniors. By examining statistics related to how well students do on particular questions, faculty can see if their students are meeting the curriculum learning objectives.

Currently, there are a few systems for assessing curricula within specific disciplines – information systems is one example (Reynolds, Longenecker, Landry, Jeffrey, Pardue, Applegate, 2004). However, what is currently lacking is a unified discipline-neutral system that everyone can potentially use. That is the impetus for our project – develop a discipline neutral system for curriculum assessment. A detailed analysis of the system is outside the scope of this paper. Rather, our goal is to shed some light on our development methodology, and how our methodology changed as we began to investigate Web 2.0. Our aim is that others might be equally motivated by what Web 2.0 has to offer.

The overall goal of our system is relatively straightforward: develop a Web-based system to facilitate program assessment. We have several sub-goals that make the project unique and challenging. First, our system is discipline-neutral, and not tied to any given profession. Second, the content of the system (questions, results, statistics, etc.) is primarily developed by the users, and not provided *a priori* by the development team. Third, we have a goal of low-maintenance, in the sense that users (at various levels of authorization) can make changes to the system without having to go through a systems administrator. Fourth, the system should be highly configurable by the user; there should be customizable ways to view, organize, and interact with the content of the system.

In late 2005, a project was begun to develop such a system. As is typical, the project began with a requirements gathering phase of user interviews in the hopes of leading into an analysis and design phase. We quickly realized that traditional systems development models were too cumbersome for the kind of project we were developing. That led us to begin creating our own methodology by investigating other development models. In the midst of this process we discovered Web 2.0.

The rest of the paper is organized as follows. Section "Web 2.0" summarizes the salient points of Web 2.0 and compares them with main-stream traditional systems development. In the "Limitations" section we describe our initial project development methodology and elaborate on the shortcomings of that methodology. The next section, "Our Experience with

Web 2.0", addresses how Web 2.0 helped to solidify a new development methodology for us and encourage us to think about new avenues for the future of the project. The final section provides some concluding remarks.

WEB 2.0

The term "Web 2.0" (O'Reilly, 2005) was coined in 2004 to refer to a set of common characteristics shared among companies that had survived the dot-com bubble burst. It can be summarized as "the Web as platform", meaning that the Web is the strategic focal point for both software development, as well as the business model. Adopting this focal point means a radical shift in the way companies view how their applications are produced, delivered, and maintained.

O'Reilly characterized Web 2.0 as a set of seven tenets. In Table 1, we summarize these tenets and compare them with more classic desktop software.

Web 2.0 Tenet	Traditional Software
Web as Platform – Offer software via a Web browser. Makes software available to everyone with a Web browser. This ensure that the "long tail" (all constituencies on the fringes of the Web) can be served by the system.	**Standalone Platform** – Offer the system as a stand-alone desktop application. The system is only available to those that have the necessary hardware and software infrastructure to run the application.
Collective Intelligence – Users contribute to the information in the system. The utility of the system increases as the number of users increase.	**Compartmentalized System** – Users of the system run it in isolation. There is no opportunity to add value to the system in cooperation with others.
Data is as Important as Computation – The system provides multi-faceted access to data (e.g., through Web Services).	**Computation is Primary** – Data is not shared between users. The system is only as good as each user contributes to it.
Perpetual Beta – The system can be improved and released every day.	**Schedule Releases** – Since the system is delivered via CD or download, releases are relatively expensive and cannot be provided daily.
Agile Development Models – Web 2.0 systems often use eXtreme (Beck and Andres, 2005) or Agile (Cockburn, 2002) methodologies to support continuous small releases.	**Sequential Development** – Although agile models are gaining in popularity, more traditional software development models still predominate.
Software Above the Level of a Single Device – Make the system available on any Web browser running on any kind of device (desktop, laptop, palmtop, TV set-top box, mobile phone, etc.)	**Proprietary Platform** – Software is typically designed for a specific operating system running on a specific kind of hardware.
Rich User Experience – Web 2.0 systems often look and behave like desktop applications, typically through the use of Ajax (Crane, Pascerello, and James, 2005).	**Static Web Pages** – Web pages essentially convey information one-way, without allowing the user to interact with the system.

Table 1 : Comparison of Web 2.0 and Traditional Software

LIMITATIONS OF OUR INITIAL DEVELOPMENT METHODOLOGY

Initially, we approached the project like a conventional systems development project following a rather typical sequential process – requirements gathering via user interviews, analysis, design, development, testing, etc. Within a month, however, we discovered several issues that led us to look for a new development methodology.

Foremost among our findings was that our vision for the assessment system meant that the project itself was going to be a substantial undertaking. So large, in fact, that the requirements gathering phase itself would take months. Typically, this would not be an issue, but the problem was compounded by the fact that we needed an initial proof-of-concept system developed within 30 days to make sure that the core of the system was feasible before continuing further. However, a traditional software development cycle does not support this process. We could certainly quickly identify those features we

wanted to see in a prototype, but other parts of the system were nebulous at best, and we could not afford to wait for the conclusion of a formal requirements-gathering phase.

After identifying a core set of features, development work on the prototype started. However, the requirements-gathering continued, and we soon discovered new ideas that we wanted to incorporate. In essence, the system became an evolving one where old ideas were tossed out, and new ideas were integrated. Some ideas we put on the "back burner", but others were essential to the core of the system. Throughout this process, development of the prototype continued in parallel.

As the prototype was being written, it quickly became apparent that some form of continual testing was necessary for the long-term success of the project. It was not enough to assume that testing would be done at some later date. Testing had to begin as soon as the first line of the prototype was written, and testing had to be continuous.

All of these issues forced us to re-evaluate the traditional system development model we were using. We wanted to do things that were not supported by existing models. In summary we needed a systems development methodology that (1) supported development of an initial prototype without a complete set of requirements, (2) was flexible enough to enable us to add new features to the system as we discovered them, (3) allowed us to differentiate those features that were critical to the system from those features that could be considered optional, and (4) supported the notion of continual testing. These shortcomings led us to investigate other development methodologies and eventually to find that Web 2.0 is an appropriate alternative development methodology that matches our goals. Furthermore, Web 2.0 offered a vision of software that altered our perception of what we wanted the system to be.

OUR EXPERIENCE WITH WEB 2.0

Our experience with Web 2.0 has affected the project on both technical and conceptual levels. On the technical level, our investigation of Web 2.0 helped us to create a coherent lightweight development methodology that is as flexible as the system we are trying to create. On the conceptual level, the tenets of Web 2.0 have validated our decision to pursue it as a development methodology, crystallized our decisions about certain aspects of the system, and stretched our thinking of what the project might be in the future.

From the beginning, we knew we wanted to develop the system for the Web. However, it wasn't until fully investigating Web 2.0 and seeing how other companies were providing access to systems via the Web did we more fully appreciate how far advanced our system might become. Although we are currently still in the prototype phase, the success of companies like Amazon and eBay validates our decision to pursue Web 2.0.

The idea of the "long tail", or serving all constituencies, has crystallized the idea that the system should potentially be usable by all faculty, programs, and schools, regardless of their size or the relative "importance" of the individuals involved. Prior to Web 2.0, this was an unvoiced assumption that we had made, but after we encountered the idea so clearly defined and articulated, we have been conscious of designing the system so that everyone interested can participate.

The idea of collective intelligence had a similar effect. We have always had the vision of users adding content to the system, but we had not fully explored the idea until faced with it in Web 2.0. Currently, we are investigating the idea that with enough people providing input into the system, the quality of the system will increase – a type of peer review process. One possible implementation would be a "test review module" where authorized users of the system could examine statistics regarding a specific question to determine whether or not the question accurately assesses knowledge of the subject. Also, we are looking at other ways of enabling users of the system to combine, organize, tag, and leverage the data in the system. For example, we are currently investigating a dynamic reporting system where users can generate statistics and reports using criteria of their own choosing, and not simply using "canned" reports provided by the system developers. Another possible extension would be a forum where users could discuss issues relevant to curriculum assessment. The experience of other Web 2.0 companies is that the more user interaction that is facilitated, the more useful the system becomes.

Google and Amazon have certainly demonstrated the concept that "data is the Intel inside" (O'Reilly, 2005), or that data is as important as computation. This has started us thinking about how we might provide access to the assessment data other than through a browser-based interface, perhaps through Web Services.

The tenet of the perpetual beta, or constant releases, was not something we had given serious thought to until discovering it described in Web 2.0. Now, our concepts of how new features will be added and how bugs will be fixed is significantly different. We now foresee the continuous release of new features to be an integral part of how the system is developed, and how the organization supports the system.

On the technical level Web 2.0 embraces an agile development methodology, exactly the kind of lightweight methodology we were seeking. Agile methods allow the system to be developed without a formal set of requirements, and provides a

method for the developers to quickly respond to requests for additional features. Combined with the idea of a perpetual beta, our system is now one of continuous evolution, rather than a fixed set of versioned releases.

Once again, the tenet of software above the level of a single device was not something we had consciously voiced. Now, however, we are currently investigating ways in which we can support multiple types of services across multiple devices. Yet again, Web 2.0 has stretched our thinking about what the project might become in the future.

Finally, the requirement of a rich user (or interactive) experience is a rather obvious one. We certainly were aware that we wanted a "nice-looking" application, but that doesn't really constitute a "rich" user experience. As a result, we are currently looking at using AJAX or other browser-based technologies to provide a more responsive web application.

CONCLUSIONS

In conclusion, Web 2.0 has been both a clarifier of our current thoughts and a motivator for the future. It has affected our project on a technical level as well as a conceptual one. It has enabled us to deliver the kind of system that we want, and to do it in a manner that makes sense to us. We do not claim that the tenets of Web 2.0 are the solution to every issue. However, we encourage readers to learn more about Web 2.0, think about how the tenets of Web 2.0 might affect their current projects, and consider incorporating discussion of Web 2.0 in their courses.

ACKNOWLEDGMENTS

The authors wish to acknowledge the contributions of Justin DeWind and Joseph Spoolstra, without whose hard work and dedication this case-study would not have been possible.

REFERENCES

1. Beck, K. and Andres, C. (2005) Extreme Programming Explained: Embrace Change, Addison-Wesley, Boston.
2. Cockburn, A. (2002) Agile Software Development, Addison-Wesley, Boston.
3. Crane, D., Pascerello, E., James, D. (2005) Ajax in Action, Manning Publications.
4. O'Reilly, T. (2005) What is Web 2.0. Design Patterns and Business Models for the Next Generation of Software, Available online at http://www.oreillynet.com.
5. Reynolds, J., Longenecker, H., Landry, J., Pardue, H., Applegate, B., (2004) Information Systems National Assessment Update: The Results of a Beta Test of the New Information Systems Exam Based on the IS 2002 Model Curriculum, *Proceedings of the Information Systems Education Journal*, vol 2, no 24, May 2004.